

SIGNAL PROCESSING STUDIO

John R. Buck and N. A. Pendergrass

ECE Department, UMass Dartmouth
285 Old Westport Rd.
North Dartmouth, MA 02747-2300
jbuck@umassd.edu, npendergrass@umassd.edu

ABSTRACT

At the University of Massachusetts Dartmouth, we teach required Junior linear systems and elective Senior signal processing courses in a studio format. This approach tightly couples short lecture segments with student exercises which include computer laboratory problems in MATLAB as well as paper and pencil analytic problems. The in-class exercises focus on issues similar to those in upcoming homework and computer project assignments. Interleaving lecture segments with problem sessions allows students to recognize immediately what they do not understand so they learn the material more quickly and effectively. Confusion can be addressed in the classroom when faculty and peer resources are at hand. The in-class exercises incorporate active and collaborative learning pedagogy. The active learning aspect requires that students take responsibility as full participants in their education. This establishes a foundation for lifelong learning. The collaborative aspect requires students to develop teamwork skills that extend outside of the classroom environment, and beyond the current semester. They become responsible for their peers' learning as well as their own.

1. ACTIVE AND COOPERATIVE LEARNING AND THE STUDIO CLASSROOM

Tinkering with most electronic products has become very difficult so students are much less likely than in the past to have had meaningful experience with any of the systems that might be discussed as an application of course material in signal processing. In our experience, this has produced a drop in the ability of students to associate theory with important practical uses. The result has often been a serious reduction of motivation to learn theory and, instead, they often depend on rote learning strategies to succeed.

In order to improve engineering education in the face of this trend, many authors have stressed the increasing importance of involving students with the practical aspects of engineering [1], [2], [3], [4], [5], [6], [7]. They have urged the use of hands-on activities that challenge students, capture their imaginations, and inspire them to want to do things with the course material. For example, Jack Lohmann of the National Science Foundation wrote "The lack of opportunities in the curricula to 'do' engineering, science, and math seriously impairs the quality of instruction and learning. The laboratories and field experiences, broadly con-

strued, are where students gain true insight." [2] Many educational researchers have also demonstrated significant improvement in motivation and information retention by using active and cooperative learning [8], [9], [10], [11], [12], [13], [14]. These methods engage and challenge students with the subject during class and get them to develop explanations and discuss issues together. These activities cause students to integrate the material into their experience which helps them to apply it to problems in later courses or practice.

Based on the educational literature and given the backgrounds of today's students with the resulting increased value of hands-on activity, we decided that the effectiveness of signal processing courses could be significantly improved by having students "do" signal processing with systems the students themselves could readily appreciate as useful. However, innovative methods and technology would be required to make activities practical because problems, experiments and projects would have to be quick and not require large amounts of faculty or technician support. Traditional laboratory and hardware methods were undesirable because considerable student and faculty effort would be directed toward laboratory and hardware problems and away from the basic signal processing concepts which are the focus of the course. Nonetheless, we wanted to have students frequently use real data so that they could fully appreciate the practical applications of signal processing.

We developed a studio classroom for signal processing courses because they have been shown to be effective and efficient for cooperative learning [15]. Our studio classroom has high speed computers with simulation software and data conversion hardware. In this classroom, students can solve real design problems but with the signals, block diagrams, concepts and mathematical descriptions covered in signal processing courses. In this paper, we specifically focus on the MATLAB studio exercises used in our Senior Digital Signal Processing elective. Previously published work [16] covers the Simulink and MATLAB exercises in our Junior year Linear Systems courses.

To understand the context of our studio class format, it is important to describe the background we assume that all senior engineering students have. They begin using MATLAB and other computer-based engineering tools as freshmen. We also assume that the students have taken a Linear Systems course, as well as its prerequisites, Differential Equations and Circuit Theory.

Due to our College of Engineering's integrated freshman

year IMPULSE program [17], [18], [19] a larger portion of the students have some experience with teaming and collaborative learning now than when we started. While helpful, this experience is not necessary. However, it is our experience that if many students do not have previous teamwork training, the instructor should expect initial resistance, and spend time developing teams and effective teaming techniques in class [8], [12].

2. STUDIO FORMAT

This section describes the typical class meeting format we use in our DSP studio course. This description contains many techniques we have found effective. However, this is not intended to be a definitive description of the only possible studio format, nor is it an exhaustive list of the effective techniques that could be used in such a format.

For students to gain the maximum benefit from a studio course, it is essential that they come to class familiar with the material to be discussed that day. To encourage them to complete the reading before arriving in class, we hold five-minute “Reading Assessment Tests” (RATs) at the start of the class meeting. These quizzes are typically four TRUE/FALSE or multiple choice questions designed such that any student who has read the assignment with moderate attention should answer all the questions correctly. A typical TRUE/FALSE RAT for an introduction to sampling class (Secs. 4.1–4.3 of [20]) is

1. It is always possible to reconstruct any continuous time signal $x_c(t)$ from samples $x[n] = x_c(nT)$ for any T .
2. If the continuous-time signal $x_c(t) = \cos(2\pi(100)t)$ is sampled with $2\pi/T = \Omega_s = 2\pi(1000)$, i.e., 1 kHz, there will be no aliasing.
3. The Nyquist sampling theorem is only valid for sinusoidal signals.
4. The discrete-time spectrum $X(e^{j\omega})$ which results from sampling $x[n] = x_c(nT)$ will always be periodic in ω with period 2π for any choice of T .

To motivate the students to do the reading, we typically use their average RAT grade over the semester as bonus points added to their homework average for the course. We differ in whether the RAT should be administered at the start of every class (JB) or roughly half the classes (NP). A quick statistical analysis of RAT scores from two semesters indicate that every student in JB’s class was sufficiently motivated by this scheme to do enough reading to perform better than chance on the RATs with $p < 0.01$.

After administering and collecting the RAT, we typically read the questions aloud and ask randomly-chosen students for the answers and some justification of why they chose that answer. These questions can be immediately addressed or brought up at an appropriate time later in class. This quick feedback reinforces several key concepts of that day’s class, and often reminds students of questions about these concepts they had while reading.

The remainder of the class time alternates between short lecture segments and group problems and projects. The lecture segments focus on the most important or most difficult

concepts in the reading. Students prefer it when these segments include a short example on the material discussed. While we typically prepare these segments in advance like traditional lectures, it is crucial to be flexible and willing to discard the planned segment in response to student’s questions. We find it helpful to think of our preparation as our *a priori* estimate of the students’ questions on the material. Like any *a priori* estimate, it should be updated or discarded in the presence of a reliable measurement of their actual questions or confusions.

Following each lecture segment, we have students work on relevant exercises in groups using one of several formats. These in-class exercises focus on concepts similar to both the example in the lecture segment and problems in the current homework assignment. Whenever it is pedagogically sensible, we attempt to construct a sequence of exercises that build through the class. We incorporate MATLAB exercises to reinforce the pencil-and-paper ones. The most successful exercises often use MATLAB exercises which verify or extend the paper exercises from earlier in the same class. Students are given a well-defined time-frame to work on the problems (typically 5 or 10 minutes). While they work, the instructor circulates between the groups observing their progress and asking and answering questions to clarify the exercise. This interaction means students immediately confront any confusion and get answers while the lecture segment and reading is still fresh in their minds. Moreover, in discussing the problems with each other, they typically refine and clarify their own understanding.

Accountability is an essential element in getting students to take the exercises seriously and work productively. Each student must produce their own written answers as a result of the group’s collaborative work. We find it effective to vary evaluation techniques to include

- Grading one randomly selected paper from each group, and giving all group members the grade of that paper,
- Requiring one randomly chosen member of a randomly chosen group to present the group’s answer orally without assistance,
- Grading all the papers from the group, then giving all group members the lowest, or the average, grade from within the group.

All of these techniques encourage collaboration and a sense of responsibility for their peers’ learning. Strong students are motivated to work with weaker students since it will help their own grades. Weaker students are generally motivated to understand rather than copy by team spirit, since oral presentations will reveal when answers were copied without understanding. All exercises not evaluated orally are graded in writing and returned at the start of the next class. Their grade on the in-class exercises is typically a part of the overall course grade commensurate with homework grades. Other more elaborate formats for team learning such as “jigsaw” exercises [12] are also effective.

Typically, we find that a RAT plus 2 or 3 lecture/exercise cycles typically fill our twice-weekly 75 minute meeting periods. In addition to the in-class RATs and exercises, we also assign weekly graded homework sets and roughly bi-weekly MATLAB projects, such as those in [21]. The MATLAB projects are often group efforts where students collab-

orate on both the programming and report writing in the same working groups used in class.

3. EXAMPLE CLASS MATERIALS

This section presents several examples of our DSP studio exercises. All of these incorporate MATLAB as a part of the in-class laboratory experience of the students in one of three pedagogical roles:

1. To check an answer obtained with pencil and paper, thus building the student's intuition. This approach teaches students to treat MATLAB as a tool to test and confirm their own answers. This avoids the common pitfall of students treating the computer as a method of obtaining an answer without engaging in critical thought about the problem or the answer given.
2. To provide a platform for numerical experiments about the properties of algorithms or systems. In this context, MATLAB provides a low overhead method to allow students to encounter real performance issues directly. Students appear to retain these lessons better than simply having an instructor say "You need to watch out for these things in this problem."
3. To give students experience with design and programming for signal processing algorithms. This is the best role, in our experience, since it requires the deepest understanding of the material, and therefore the best opportunity for learning.

A pedagogically attractive aspect of the second and third roles is that they allow the students to make instructive mistakes in class. We find that students often remember the material better for making these mistakes and correcting them, than if they get it right the first time, or become frustrated by the mistakes when working alone.

We present examples of MATLAB exercises illustrating the different pedagogical roles below.

3.1. Linear Phase

This example required students to use MATLAB to check their answers and develop intuition about the relationships between impulse response symmetry, linear phase, and constant group delay. Following the RAT, the class began with a short lecture (1.5 pages of lecture notes) covering the relationship between symmetry in $h[n]$, linear phase in $H(e^{j\omega})$, and constant group delay. The students were then given 10 minutes to work in groups on the questions below, and were told that one randomly chosen member from each group would have to answer one of the questions orally and justify the group's answer. The entire group received their grade based on that person's answers. In the interest of space, we omitted the plots of the four impulse responses referred to in the problem. They were all similar in nature to those given in Problem 5.19 of [20]. This problem also stimulated students thinking about issues of group delay in preparation for their upcoming MATLAB homework project on phase effects of lowpass filters (Project 6.4 in [21].)

Problem 1 Without calculating $H(e^{j\omega})$ explicitly, give the phase $\angle H(e^{j\omega})$ and group delay $\text{grd}\{H(e^{j\omega})\}$ for each

of the LTI systems whose impulse responses $h[n]$ are shown below.

After the students presented their answers orally, there was a short 5 minute lecture on the use of `grpdelay` to find and plot the group delay of a system from its impulse response. The students were then given 5 minutes to check their earlier answers. Students had previous experience using `filter` and `freqz` to specify a system a finite impulse response.

3.2. Operations counting for the FFT

This exercise used MATLAB as an experimental platform. After a relatively long lecture (4 pages of notes) on the Decimation-in-Frequency FFT algorithm based on Section 9.4 of [20], the students were given the following problem and told to work on it in their groups for 15 minutes. In addition, they were told that one randomly selected written answer from each group would be graded to provide the entire group's grade.

Problem 2 In this problem, you will look at the amount of computation MATLAB requires to compute an FFT for different lengths N . To measure computation, you will use the MATLAB `flops` command, which counts the total number of Floating Point Operations, or flops. To reset this counter, use the command `flops(0)`. To get the current value of the counter, type `flops`. For example

```
>> flops(0)
>> 2*2+3
ans =
     7
>> flops
ans =
     2
```

As we would expect, evaluating `2*2+3` requires two operations, one multiply and one add, for a total of two flops.

- (a) Use `randn` to define `x1021` to be a random signal of length $N = 1021$. Use `flops` to determine how many floating point operations are required to compute the 1021 point DFT of this signal using `fft`.
- (b) Reset `flops` with `flops(0)` before proceeding to the next part.
- (c) Use `randn` to define `x1024` to be a random signal of length $N = 1024$. Use `flops` to determine how many floating point operations are required to compute the 1024 point DFT of this signal using `fft`. How does this compare to the number of flops required to find the DFT of `x1021`? Does this make sense. Explain your results in a short 2-3 sentence paragraph.

3.3. Block Convolution

This programming example was used in a class on block convolution using the FFT. Specifically, the class began with a moderate lecture (2 pages of notes) on the Overlap-Add Technique for block convolution. Students were then given the remainder of the class to solve the two problems listed

below and hand in their results. This class was relatively late in the term, so they were experienced at working together, and with MATLAB. In our experience, it would be difficult early in the semester to get them to work in a focused manner for such an extended period of time. During the exercise, the instructor circulated continuously among the groups helping them with questions.

Problem 3 In this problem, you will complete a MATLAB function to implement Overlap-add block convolution. Most of the function has already been written for you. You are required to fix four lines. These lines have symbols like <XXXX> in them and the comment line % FIX THIS just above them.

- Make a local copy of the function `olapadd.m` from the `Class22` folder on the ECE 475 Section of the Server. Complete the function.
- Make a local copy of the data file `c122data.mat` from the ECE475 Server, and load this data.
- Compute the convolution of the data signal `x` and the filter with impulse response `h65` using `conv`. Save this result in the variable `yconv`.
- Define `yola` to be the result of convolving `x` with `h65` using the overlap add function you defined and a block size of $N = 16384$.
- If your version of `olapadd.m` is correct, the signals `yconv` and `yola` should agree closely. Compute the difference between the two signals, and verify that the largest absolute value of the error signal is less than 10^{-12} . Note that `yola` is longer than `yconv`, so you will have to ignore several points at the end of `yola` to do the subtraction.

The function mentioned, `olapadd.m`, is given below:

```
function y = olapadd(x,h,N)
% y = olapadd(x,h,N) computes the convolution
% of x[n] with h[n] using the overlap add
% method of block convolution. the block
% convolutions are done with a block of length
% N, which must be at least twice the length
% of h[n].
P = length(h);
if (N<(2*P))
    error(['Block size N must be at least' ...
        ' twice the length of h[n]']);
end
% find block size for x[n] so that the
% circular convolution gives the same result
% as the linear convolution. you have already
% been told what N and P are, so now you must
% figure out how big the blocks of x[n] should
% be.
% FIX THIS
L = <XXXX>;

% find dft of impulse response once and reuse
% it with each block.
H = fft(h,N);
```

```
% compute how many blocks required
nblocks = ceil(length(x)/L);
% if the last block is not an a full block,
% zero pad x to make it an even number of
% blocks long.
if ((nblocks*L) ~= length(x))
    x = [x zeros(1,nblocks*L-length(x))];
end

% set up empty vector to store results of each
% block convolution.
y = zeros(1,length(x)+P-1);

for block = 1:nblocks
% extract the current input block from the
% overall signal.
% FIX THIS
xblock = x(<YYYY>);
% compute the DFT of the current input block.
Xblock = fft(xblock,N);
% compute the DFT of the current output block.
% FIX THIS
Yblock = <ZZZZ>;
% find the time signal yr[n] from the DFT.
yblock = ifft(Yblock);
% Put the output block into the overall
% output signal.
% FIX THIS
y(<QQQQ>) = y(<QQQQ>) + yblock;
end
```

The second problem in this class was an experiment, building on their knowledge of the `flops` counter from the earlier `fft` class.

Problem 4 For this problem, you will examine the speed of different methods of convolving a large signal with a filter.

- Use the `flops` command to find how many operations it requires to compute the convolution of `x` with `h65` using both the `conv` command and the `olapadd` command you just wrote. For a 65 point filter, which is the most efficient method of computing the convolution?
- Use the `flops` command to find how many operations it requires to compute the convolution of `x` with `h1023` using both the `conv` command and the `olapadd` command you just wrote. For a 1023 point filter, which is the most efficient method of computing the convolution?
- The two filters `h65` and `h1023` are both lowpass filter with a cutoff of $\omega_c = \pi/2$. If you compute the output using the `conv` command, how does the amount of computation required change when the length of the filter goes from 65 points to 1023 points? If you use `olapadd`, how does the amount of computation change as the filter length gets longer? Explain your answers clearly.

3.4. Discrete-time Processing of Continuous-Time Signals

This design example was taken from a class covering the use of discrete-time systems and sampling to process continuous-time signals. The reading material for the class covered the relationship between the frequency response of the discrete-time system $H(e^{j\omega})$ and the continuous-time frequency response of the overall system $H(j\Omega)$, e.g., Section 4.4 of [20]. The class began with a short lecture (1 page) covering this material, including an example of how the discrete-time filtering impacts the continuous-time spectra (Fig. 4.13 of [20]). The students were then given 20 minutes to work in groups to complete the exercise and write up their answers for grading. This exercise used the students' previous experience using `fft`, `butter`, `filter`, and a local function `mat2wav` (a slightly improved `wavwrite`).

Problem 5 Let $1/T = 48000$ Hz. Define `x` in MATLAB to be samples of $x_c(t) = \cos(2\pi(1000)t) + \cos(2\pi(8000)t)$ for $0 \leq t \leq 1$ s.

- Plot $|X(e^{j\omega})|$ using `fft` as in the last class. Are the peaks at the values of ω where you expect them?
- Use `butter` to design a 6th order lowpass filter $H(e^{j\omega})$ that will keep the 1000 Hz signal but remove the 8000 Hz signal.
- Filter `x` to get an output `y` using the `filter` command with the `a` and `b` coefficients given by `butter`.
- Look at $|Y(e^{j\omega})|$ using `fft` to confirm that your filter worked as intended.
- Save `x` and `y` as `.wav` files and listen to both. Can you hear the difference you expected?

4. SUMMARY

We have found active and collaborative learning techniques in a studio setting to be an effective pedagogical approach to teaching DSP in a Senior elective. We hope that in presenting these examples we will spur further development of these ideas by other educators in the field.

REFERENCES

- [1] Carnegie Mellon University. Designing a Curriculum for the '90's, Why Change? *IEEE Currents*, pages 4–5, Winter 1992.
- [2] J. R. Lohmann. Myths, facts and the future of U.S. engineering and science education. *ASEE J. Eng. Educ.*, pages 365–371, April 1991.
- [3] Joseph Bordogna, Eli Fromm, and Edward W. Erust. Engineering education: Innovation through integration. *ASEE J. Eng. Educ.*, pages 3–8, January 1993.
- [4] Robert G. Quinn. Drexel's E4 program: A different professional experience for engineering students and faculty. *ASEE J. Eng. Educ.*, pages 196–202, October 1993.
- [5] Robert G. Quinn. The E4 introductory test, design and simulation laboratory. *ASEE J. Eng. Educ.*, pages 223–226, October 1993.
- [6] Bonney H. Sheahan and John A. White. Quo Vadis, undergraduate engineering education. *J. Engineering Education*, pages 1017–1022, December 1990.
- [7] Jeff Meade. Change is in the wind. *ASEE Prism*, pages 20–24, May 1993.
- [8] D. Johnson, R. Johnson, and K. Smith. *Active Learning: Cooperation in the College Classroom*. Interaction Book Co., 1998.
- [9] V. Ercolano. Learning through cooperation. *ASEE Prism*, pages 26–29, November 1994.
- [10] R. Hake. Interactive-engagement vs. traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses. *Am. J. Phys.*, 66(64), 1998.
- [11] R. Schwartz. Working together to succeed. *ASEE Prism*, pages 31–34, March 1996.
- [12] R. Felder and R. Brent. Cooperative learning in technical courses: Procedures, pitfalls, and payoffs. Technical Report ED377038, ERIC Document Reproduction Service, 1994.
- [13] R. Dees. The role of cooperative learning in increasing problem solving ability in a college remedial course. *J. Research In Math. Educ.*, 22(5):409–421, 1991.
- [14] M. Lumsdaine and E. Lumsdaine. Thinking preferences of engineering students: Implications for curriculum restructuring. *ASEE J. Eng. Educ.*, pages 194–204, April 1995.
- [15] Charles M. Vest. The transformation of engineering education. *Syllabus, Engineering & Science*, pages 2–5, Spring 1993. Available at http://209.134.33.92/archive/EngSci/Syll/EngSci_1/Transform_of_Eng_in_Ed.txt.
- [16] N. A. Pendergrass. Using computers, simulators and sound to give hands-on experience. In *Proceedings of the ASEE National Conference*, June 1996.
- [17] N. A. Pendergrass, Raymond N. Laoulache, John P. Dowd, and Robert E. Kowalczyk. Efficient development and implementation of an integrated first year engineering curriculum. In *Proceedings of the Frontiers In Education Conference*, November 1998.
- [18] N. A. Pendergrass, Robert E. Kowalczyk, John P. Dowd, Raymond N. Laoulache, William Nelles, James A. Golen, and Emily Fowler. Improving first year engineering education. In *Proceedings of the Frontiers In Education Conference*, San Juan, PR, November 1999.
- [19] N. A. Pendergrass, Raymond N. Laoulache, and Paul J. Fortier. Mainstreaming an innovative 31-credit curriculum for first-year engineering majors. In *Proceedings of the Frontiers In Education Conference*, Kansas City, MO, October 2000.
- [20] Alan V. Oppenheim and Ronald W. Schaffer with John R. Buck. *Discrete-time Signal Processing*. Prentice Hall Signal Processing Series. Prentice Hall, Englewood Cliffs, NJ, second edition, 1999.

- [21] J. R. Buck, M. Daniel, and A. C. Singer. *Computer Explorations in Signals and Systems Using Matlab*. Matlab Curriculum Series. Prentice Hall, Englewood Cliffs, NJ, 1997.